BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pakar

Secara umum, Sistem Pakar *(expert system)* adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang biasa dilakukan oleh para ahli Sistem Pakar yang baik dirancang agar dapat menyelesaikan suatu permasalahan tertentu dengan meniru kerja para ahli. Dengan Sistem Pakar ini, orang awam juga dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli. Bagi para ahli, Sistem Pakar ini juga akan membantu aktivitasnya sebagai asisten yang sangat berpengalaman (Rahayu ; 2013 : 130).

II.2. Karakteristik Sistem Pakar

II.2.1. Sistem Pakar

Ada beberapa definisi tentang Sistem Pakar, antara lain sebagai berikut :

1. Menurut Durkin : Sistem Pakar adalah suatu program komputer yang dirancang untuk memodelkan kemampuan penyelesaian masalah yang dilakukan seorang pakar.
2. Menurut Ignizo : Sistem Pakar adalah suatu model dan prosedur yang berkaitan, dalam suatu domain tertentu, yang mana tingkat keahliannya dapat dibandingkan dengan keahlian seorang pakar.

3. Menurut Giarratano dan Riley : Sistem Pakar adalah suatu sistem computer yangbisa menyamai atau meniru kemampuan seorang pakar (Nasrullah ; dkk ; 2013 : 2-3).

II.2.2. Bentuk Sistem Pakar

Ada 4 bentuk Sistem Pakar, yaitu sebagai berikut : 1. Berdiri sendiri, Sistem Pakar jenis ini merupakan *Softwere* yang berdiri sendiri, tidak tergantung dengan *softwere* yang lain. 2. Tergabung. Sistem Pakar jenis ini merupakan bagian dari program yang terkandung di dalam suatu algoritma (konvensional) atau merupakan program dimana di dalamnya memanggil algoritma subrutin lain (konvensional). 3. Menghubungkan dengan *softwere* yang lain. Bentuk ini biasanya menghubungkan Sistem Pakar yang menghubungkan suatu paket program tertentu, misalnya DBMS. 4. Sistem Mengabdi. Sistem Pakar merupakan bagian dari komputer khusus yang dihubungkan dengan suatu fungsi tertentu, misalnya Sistem Pakar yang digunakan untuk membantu menganalisis data radar (Nasrullah ; dkk ; 2013 : 3).

Ada beberapa alasan mendasar mengapa sistem pakar dikembangkan untuk menggantikan seorang pakar, di antaranya :

1. Dapat menyediakan kepakaran setiap waktu dan di berbagai lokasi.
2. Secara otomatis mengerjakan tugas-tugas rutin yang membutuhkan seorangpakar.
3. Seorang pakar akan pensiun atau pergi.
4. Seorang pakar adalah mahal.

5. Kepakaran juga dibutuhkan padalingkungan yang tidak bersahabat (*hostile environtment*) (Nasrullah ; dkk ; 2013 : 3).

II.3. Penyakit *Kanker Nasofaring*

Karsinoma Nasofaring (KNF) merupakan tumor ganas yang paling banyak dijumpai diantara tumor ganas THT di Indonesia, dimana KNF termasuk dalam lima besartumor ganas, dengan frekuensi tertinggi (bersama tumor ganas serviks uteri, tumor payudara, tumor getah bening dan tumor kulit), sedangkan didaerah kepala dan leher menduduki tempatpertama KNF mendapat persentase hampir 60% dari tumor di daerah kepala dan leher, diikuti tumor ganas hidung dan sinus paranasal 18%, laring 16%, dan tumor ganas rongga mulut, tonsil, hipofaring dalam persentase rendah.

Penyakit kanker *nasofaring* memiliki 4 stadium penyakit. stadium penyakit I dan II termasuk stadium terendah dan stadium III dan IV adalah stadium tertinggi. (Wulan Melani, Ferryan Sofyan, 2013; 2).

II.4. Metode *Certainty Factor*

Faktor kepastian (certainty factor) menyatakan kepercayaan dalam sebuah kejadian (fakta atau hipotesa) berdasar bukti atau penilaian pakar. Certainty factor menggunakan suatu nilai untuk mengasumsikan derajat keyakinan seorang pakar terhadap suatu data.

CF(Rule) = MB[H,E] – MD[H,E] .....(II.1)

$$MB(H,E) = \begin{cases} 1 & P(H) = 1 \\ \dfrac{\max[P(H|E),P(H)] - P(H)}{\max[1,0] - P(H)} & \text{lainnya} \end{cases} \quad \text{... ... ... (II.2)}$$

$$MD(H,E) = \begin{cases} 1 & P(H) = 0 \\ \dfrac{\min[P(H|E),P(H)] - P(H)}{\min[1,0] - P(H)} & \text{lainnya} \end{cases} \quad \text{... ... ... (II.3)}$$

Dimana :

CF(Rule) =Faktor kepastian

MB(H,E) =*measure of belief* (ukuran kepercayaan) terhadap hipotesis H, jika diberikan *evidence* E (antara 0 dan 1)

MD(H,E)=*measure of disbelief* (ukuran ketidakpercayaan) terhadap *evidence*H, jika diberikan *evidence* E (antara 0 dan 1)

P(H) = Probabilitas kebenaran hipotesis H

P(H|E) = Probabilitas bahwa H benar karena fakta E

Adapun logika metode certainty factor pada sesi konsultasi sistem, pengguna konsultasi diberi pilihan jawaban yang masing-masing memiliki bobot, dapat dilihat pada tabel II.1.

Tabel II.1.Tabel Nilai User

Keterangan	Nilai User
Tidak	0
Tidak tahu	0.2
Sedikit yakin	0.4
Cukup yakin	0.6
Yakin	0.8
Sangat yakin	1

Nilai 0 menunjukkan bahwa pengguna konsultasi bahwa pengguna konsultasi menginformasikan bahwa user tidak mengalami gejala seperti yang ditanyakan oleh sistem. Semakin pengguna konsultasi yakin bahwa gejala tersebut memang dialami manusia, maka semakin tinggi pula hasil persentase keyakinan yang diperoleh. Proses perhitungan presentase keyakinan diawali dengan pemecahan sebuah kaidah yang premis majemuk, menjadi kaidah-kaidah yang memiliki premis tunggal. Kemudian masing-masing aturan baru dihitung certainty factornya. (Nur Anjas Sari; 2013; 2).

II.5. Metode *Dempster Shafer*

Dempster-Shafer adalah suatu teori matematika untuk pembuktian berdasarkan *belief functions and plausible reasoning* (fungsi kepercayaan dan pemikiran yang masuk akal), yang digunakan untuk mengkombinasikan potongan informasi yang terpisah (bukti) untuk mengkalkulasi kemungkinan dari suatu peristiwa.Teori ini dikembangkan oleh Arthur P. Dempster dan Glenn Shafer. Secara umum teori Dempster-Shafer ditulis dalam suatu interval :

[*Belief,Plausibility*].....(II.8)

1) Belief (*Bel)* adalah ukuran kekuatan evidence dalam mendukung suatu himpunan proposisi. Jika bernilai 0 maka mengindikasikan bahwa tidak ada evidence, dan jika bernilai 1 menunjukkan adanya kepastian. Dimana nilai bel yaitu(0-0.9).
2) *Plausibility* (Pl) dinotasikan sebagai :

   Pl(s) = 1 – Bel (-s) .....(II.9)

Plausibility juga bernilai 0 sampai 1. Jika yakin akan -s, maka dapat dikatakan bahwa Bel(-s)=1, dan Pl(-s)=0.

Padateori *Dempster-Shafer* dikenal adanya *frame of discrement* yang dinotasikan dengan θ. Frame ini merupakan semesta pembicaraan dari sekumpulan hipotesis. Tujuannya adalah mengaitkan ukuran kepercayaan elemen-elemen θ.Tidak semua evidence secara langsung mendukung tiap-tiap elemen. Untuk itu perlu adanya probabilitas fungsi densitas (m). Nilai tidak hanya mendefinisikan elemen-elemen θ saja, namun juga semua subsetnya. Sehingga jika θ berisinelemen,maka sub set θ adalah 2*n* .Jumlah semua m dalam subset θ sama dengan 1. Apabila tidak ada informasi apapun untuk memilih hipotesis, maka nilai :

$$m\{\theta\} = 1{,}0.$$

Apabila diketahui X adalah subset dari θ, dengan m 1 sebagai fungsi densitasnya, dan Y juga merupakan sub set dari θ dengan m2 sebagai fungsi densitasnya,maka dapat dibentuk fungsi kombinasi m1 dan m2 sebagai m3, yaitu (Muhammad Dahria, et al., 2013) :

$$m_3(Z) = \frac{\sum X \cap Y = Z \, m_1(X).m_2(Y)}{1 - \sum X \cap Y = \emptyset \, m1\,(X).m2(Y)} \quad \text{(II.10)}$$

II.6. *Microsoft Visual Basic 2010*

Microsoft Visual Basic .NET adalah sebuah alat untuk mengembangkan dan membangun aplikasi yang bergerak di atas system .*NET Framework,* dengan menggunakan bahasa BASIC. Dengan menggunakan alat ini, para *programmer* dapat membangun aplikasi *Windows Forms*, Aplikasi web berbasis ASP .NET,

dan juga aplikasi *command line*. Alatini dapat diperoleh secara terpisah dari beberapa produk lainnya (sepert i*Microsoft Visual C++, Visual C #,* atau *Visual J#*), atau juga dapat diperoleh secara terpadu dalam *Microsoft Visual Studio .NET*. Bahasa *Visual Basic .NET* sendiri menganut paradigm bahasa pemrograman berorientasi objek yang dapat di lihat sebagai evolusi dari *Microsoft Visual Basic* versi sebelum nya yang di implementasikan diatas *.NET Framework*. Peluncurannya mengundang kontroversi, mengingat banyak sekali perubahan yang dilakukan oleh *Microsoft*, dan versi baru ini tidak kompatibel dengan versi terdahulu. Teknologi *.NET* muncul karena adanya beberapa alasan pada teknologi aplikasi *Microsoft* yang lama (Ahmad Rais Ruli, 2017 : 11).

VB.NET mempunyai beberapa fitur baru yang membuat bahasa VB menjadi lebih kuat sehingga dapat mematahkan mitos bahwa VB hanya bahasa mainan (*toy language*) bila dibandingkan dengan bahasa lain seperti C++ dan Java. Fitur-fitur tersebut antara lain:

1. Dukungan *Object Oriented Programming*

   VB.NET adalah bahasa pemrograman yang penuh *Object Oriented*.Jadi *VB.NET* mendukung fitur-fitur OOP seperti *inheritance*, *interface*, *method overloading* dan *polymorphism*.

2. *Structure Exception Handling*

   Untuk menggantikan perintah *On Error Goto* pada VB6, *VB.NET* meyediakan *error handling* yaitu *Try..Catch..Finally*. *Error handling* pada *VB.NET* ini lebih mudah digunakan karena hanya cukup menaruh kode yang

akan dicek dibagian *Try*, dan menyiapkan *exception handling* nya dibagian *catch*.

3. GDI+

   GDI+ adalah *library* grafik yang digunakan untuk mengembangkan aplikasi *windows form*.

4. *Web Services* dan *Web Form*

   Dengan penggunaan *VB.NET* kita dapat membuat aplikasi berbasis web dengan menggunakan *Web Form* (ASP.NET). Kita juga dapat membuat aplikasi *web service* untuk membuat *three tier application*.

5. *Cross-Languege Interoperability*

   Oleh karena disetiap program yang berjalan di *.NET* dikompile menjadi *assembly* maka dapat dibuat aplikasi dengan bahasa pemrograman yang berbeda yang berjalan diatas *platform .NET* seperti C# dan C++.Jadi dapat juga menggunakan komponen yang dibuat menggunakan C# atau C++ untuk digunakan di VB.

6. *Multihreading*

   Fitur ini sangat berguna apabila terdapat aplikasi yang proses komputasinya memakan waktu lama.

7. *Type Safe Collection*

   Fitur ini mulai ada di .NET 2.0 (VB 8 atau VB 2005). Dengan fitur ini dapat dibuat *object collection*yang *type safe*.

8. LINQ (*Language Integrated Query*)

   Fitur ini mulai ada pada .NET 3.5 (VB 9 atau VB 2008). Fitur ini digunakan untuk dilakukan *query* data yang ditambahkan kedalam bahasa VB dan C# sehingga kedua bahasa tersebut dapat melakukan *query* ke objek *database* (Ahmad Rais Ruli, 2017 : 11-12).

II.7. SQL Server 2008

Microsoft SQL Server adalah sebuah system manajemen basis data relasional (RDBMS) produk*Microsoft*. Bahasa quer iutama nya adalah *Transact-SQL* yang merupakan implementasi dari *SQL* standar ANSI/ISO yang digunakan oleh *Microsoft* dan *Sybase*. Umumnya *SQL Server* digunakan di dunia bisnis yang memiliki basis data berskala kecil sampai dengan menengah, tetapi kemudian berkembang dengan digunakannya *SQL Server* pada basis data besar.

SQL Server 2008 menyimpan data dengan konsep *Relational Database*. Selain itu penyajianya merupakan penyajian pada level fisik karena kita akan langsung menyimpan data pada *database* dengan kondisi yang sebenarnya, yaitu di simpan pada tabel apa, kolom nama, dan menggunakan tipe data apa saat penyimpanan (Ilham Akbar, et al., 2013 : 90).

II.8. *Database*

Database sering didefinisikan sebagai kumpulan data yang terkait.Secara teknis, yang berada dalam sebuah database adalah sekumpulan tabel atau objek

lain (indeks, view, dan lain-lain).Tujuan utama pembuatan database adalah untuk memudahkan dalam mengakses data (Eka Choliviana dan Lies Yulianto, 2013).

Database adalah sekumpulan data yang berisi informasi mengenai satu atau beberapa object.Data dalam database tersebut biasanya disimpan dalam tabel yang saling berhubungan antara satu dengan yang lain (Dani Ainur Rivai dan Sukardi, 2013).

II.9. Normalisasi

Normalisasi adalah suatu teknik dengan pendekatan *bottom-up* yang digunakan untuk membantu mengidentifikasikan hubungan. Dimulai dari menguji hubungan, yaitu *functional* dependencies antara atribut.Pengertian lainnya adalah suatu teknik yang menghasilkan sekumpulan hubungan dengan sifat-sifat yang diinginkan dan memenuhi kebutuhan pada perusahaan. (Indrajani : 2015 ; 7).

Adapun bentuk-bentuk normalisasi menurut (Indrajani : 2015 ; 9-10) adalah sebagai berikut:

1. Unnormalized Form (UNF)

   Merupakan suatu tabel yang berisikan satu atau lebih grup yang berulang. Membuat tabel yang *unnormalized*, yaitu dengan memindahkan data dari sumber informasi.

   Contoh: nota penjualan yang disimpan ke dalam format tabel dengan baris dan kolom.

2. First Normal Form (1NF)

   Merupakan sebuah relasi di mana setiap baris dan kolom berisikan satu dan hanya satu nilai.

   Proses UNF ke 1NF

   a. Tentukan satu atau sekumpulan atribut sebagai kunci untuk tabel *unnormalized*.
   b. Identifikasikan grup yang berulang dalam tabel *unnormalized* yang berulang untuk kunci atribut.

3. Second Normal Form (2NF)

   Berdasarkan pada konsep *full functional dependency*, yaitu A dan B merupakan atribut sebuah relasi. B dikatakan *fullydependent* terhadap A jika B *functional dependent* pada A tetapi tidak pada propersubset dari A. 2NF merupakan sebuah relasi dalam 1NF dan setiap atribut *non-primary-key* bersifat *fully functionally dependent* pada primary key.

   1NF ke 2NF

   a. Identifikasikan *primary key* untuk relasi 1NF.
   b. Identifikasikan *functional dependencies* dalam relasi.
   c. Jika terdapat *partial dependencies* terhadap *primary key*, maka hapus dengan menempatkan dalam relasi yang baru bersama salinan determinannya.

4. Third Normal Form (3NF)

   Berdasarkan pada konsep *transitive dependency*, yaitu suatu kondisi di mana A, B, dan C merupakan atribut sebuah relasi, maka A – B dan B – C, maka

transitively dependent pada A melalui B. 3NF adalah sebuah relasi dalam 1NF dan 2NF, di mana tidak terdapat atribut *non primary key* yang bersifat *transitivelydependent* pada *primary key*.

2NF ke 3NF

a. Identifikasi primary key dalam relasi 2NF.
b. Identifikasi *functional dependencies* dalam relasi.
c. Jika terdapat *transitivedependencies* terhadap *primary key* hapus dengan menempatkannya dalam relasi yang baru bersama dengan salinan determinannya.

5. Boyce-code Normal Form (BCNF)

Berdasarkan pada *functional dependencies* yang dimasukkan dalam hitungan seluruh *candidate key* dalam suatu relasi. Bagaimanapun BCNF juga memiliki batasan-batasan tambahan disamakan dengan defenisi umum dari 3NF. Suatu relasi dikatakan BCNF, jika dan hanya jika setiap determinan merupakan *candidate key*. Perbedaan antara 3NF dan BCNF yaitu untuk *functional dependent* A – B, 3NF memungkinkan *dependency* ini dalam suatu relasi jika adalah atribut *primary key* dan A bukan merupakan *candidate key*. Sedangkan BCNF menetapkan dengan jelas bahwa untuk *dependency* ini agar ditetapkan dalam relasi A, maka A harus merupakan *candidate key*. Setiap relasi dalam BCNF juga merupakan 3NF, tetapi relasi dalam 3NF belum tentu termasuk ke dalam BCNF. Dalam BCNF kesalahan jarang sekali terjadi, Kesalahan dapat terjadi pada relasi yang (Indrajani : 2015 ; 9-10):

a. Terdiri atas 2 atau lebih *composite candidate key*.

b. *Candidate key overlap*, sedikitnya satu atribut.

II.10. *Unified Modelling Language* (UML)

Unified Modeling Language (UML) merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa pemodelan visual yang memungkinkan pengembang system membuat *blue print*atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui jumlah elemen grafis yang bisa dikombinasikan menjadi diagram (Rosana Junita Sirait, et al., 2015).

Bahasa pemodelan UML lebih cocok untuk pembuatan perangkat lunak dalam bahasa pemrograman berorientasi objek (C+,Java,VB.NET), namun demikian tetap dapat digunakan pada bahasa pemrograman prosedural.

Berdasarkan beberapa pendapat di kemukakan di atas dapat ditarik kesimpulan bahwa "*Unified Modeling Language* (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk menvisualisasikan, menspesifikasikan ,membangun dan pendokumentasian dari sebuah system pengembangan perangkatlunak berbasis OO (*Object Oriented*) (Aris, et al., 2015).

II.10.1.Fungsi *Unified Modelling Language* (UML)

Unified Modeling Language (UML) biasa digunakan untuk (Aris, et al., 2015) :

1. Menggambarkan batasan sitem dan fungsi-fungsi system secara umum, dibuat dengan *use case* dan *actor*.
2. Menggambarkan kegiatan atau proses bisnis yang dilaksanakan secara umum, dibuat dengan *interaction diagrams*.
3. Menggambarkan representasi struktur *static* sebuah system dalam betuk *class diagrams*.
4. Membu at model behavior “yang menggambarkan kebiasaan atau sifat sebuah sistem” dengan *state transition diagrams*.
5. Menyatakan arsitektur implementasi fisik menggunakan *component and development.*
6. Menyampaikan atau memperluas *fungsionality* dengan *stereotypes*.

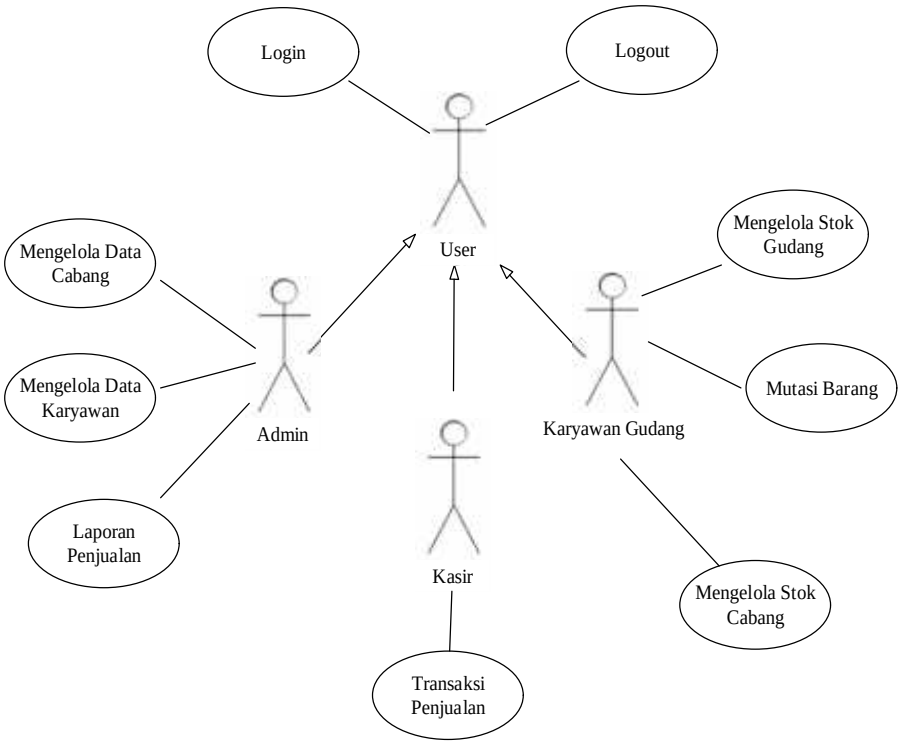
UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan system berorientasi objek. Karena UML menyediakan bahasa pemodelan visual yang memungkinkan pengembang sistem membuat *blue print* atas visi nya dalam bentuk yang baku. UML berfungsi sebaga ijembatan dalam mengkomunikasikan beberapa aspek dalam sistem melalui sejumlah elemen grafis yang bisa di kombinasikan menjadi diagram. UML mempunyai banyak diagram yang dapat mengakomodasikan berbagai sudut pandang dari suatu perangkat lunak yang akan di bangun. Diagram-diagram tersebut di gunakan untuk :

1. Mengkomunikasikan ide.
2. Melahirkan ide-ide baru dan peluang-peluang baru.

diagram digunakan untuk penggambaran *use case* statik dari suatu sistem. *Use case* diagram penting dalam mengatur dan memodelkan kelakuan dari

suatu sistem. *Use case* menjelaskan apa yang dilakukan system (atau sub sistem) tetapi tidak menspesifikasi cara kerjanya. *Flow of event* digunakan untuk menspesifikasi cara kerjanya kelakuan dari *use case*. *Flow of event* menjelaskan *use case* dalam bentuk tulisan dengan sejelas-jelasnya, dintaranya bagaimana, kapan *use case* dimulai dan berakhir, ketika *use case* berinteraksi dengan aktor, objek apa yang digunakan, alur dasar dan alur alternatif. Terdapat beberapa symbol dalam menggambarkan diagram *use case*, yaitu *use cases*, aktor dan relasi (Achmad Hamzah Nasrullah dan Dadang Sudrajat, 2015).

Adapun contoh Use Case Diagram dapat dilihat pada gambar II.1.

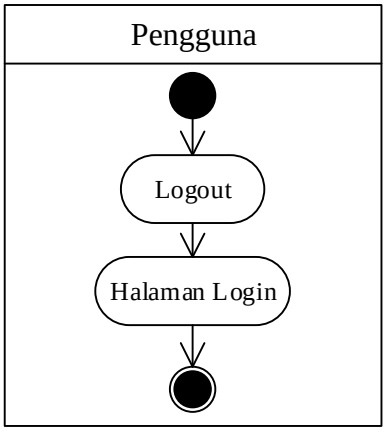


Gambar II.1. *Use Case* Diagram
(Sumber :Ade Hendini, 2016 : 112)

1. *Activity* Diagram

Activity diagram memperlihatkan alur langkah demi langkah dalam suatu proses. Suatu aktivitas menunjukkan sekumpulan aksi (secara sekuensial atau bercabang dari satu aksi ke aksi lain), dan nilai yang dihasilkan atau digunakan oleh aksi-aksi yang terjadi. Activity diagram ditunjukkan untuk memodelkan fungsi dari suatu system dan menekankan pada alur dari control didalam pelaksanaan dari suatu tindakan (Achmad Hamzah Nasrullah dan Dadang Sudrajat, 2015).

Adapun contoh Activity Diagram dapat dilihat pada gambar II.2.



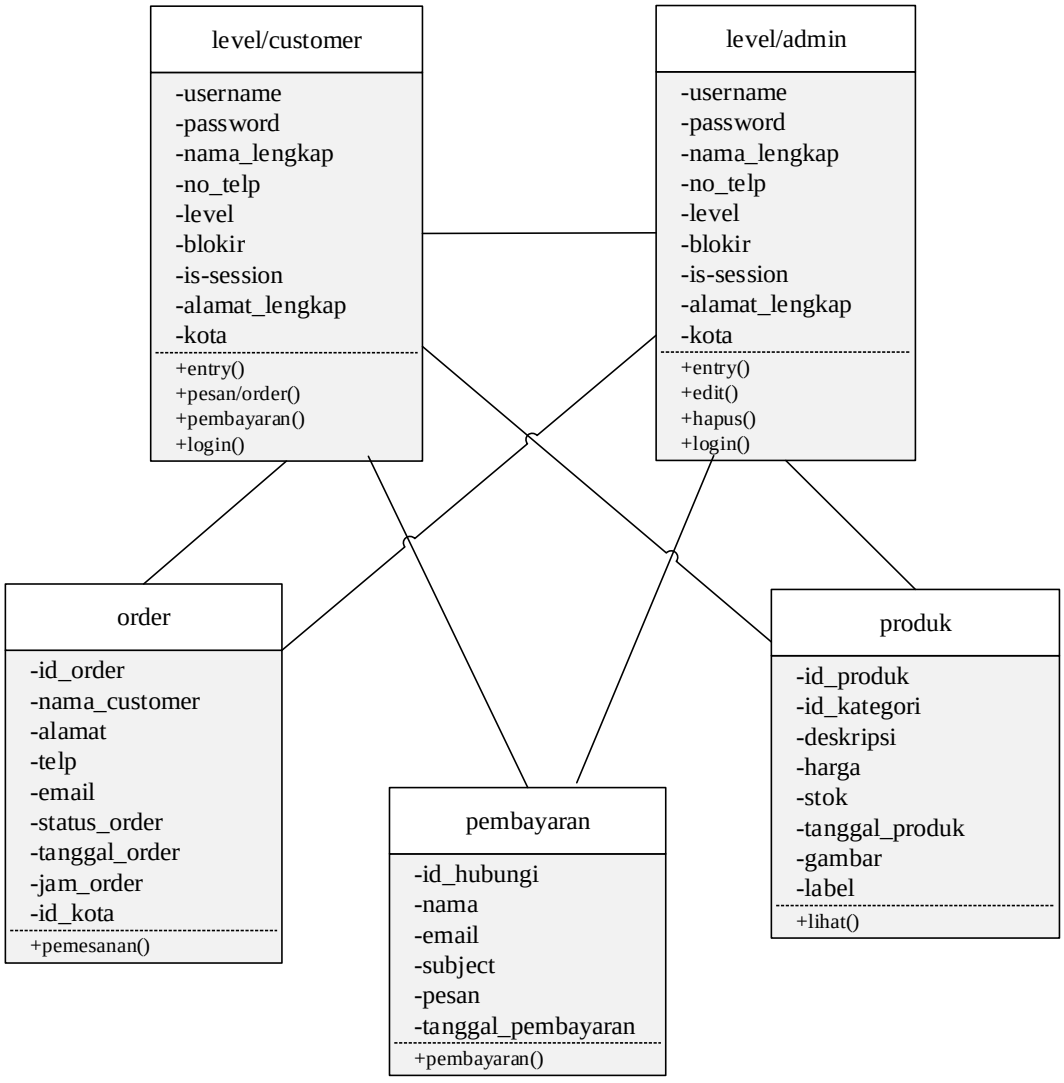
Gambar II.2. *Activity* Diagram
(Sumber :Ade Hendini, 2016 : 112)

2. *Class* Diagram

Class diagram menunjukan sekumpulan kelas, antar muka, dan kerja sama serta hubungannya. *Class* diagram digunakan untuk memodelkan perancangan static dari gambaran sistem. Biasanya meliputi pemodelan *vocabulary* dari sistem, pemodelan kerjasama, atau pemodelan skema. *Class*

diagram dapat digunakan untuk membangun system yang dapat dieksekusi melalui teknik *forward and reverse*, selain untuk penggambaran, menspesifikasikan, dan pendokumentasian struktur model (Achmad Hamzah Nasrullah dan Dadang Sudrajat, 2015).

Adapun contoh Class Diagram dapat dilihat pada gambar II.3.

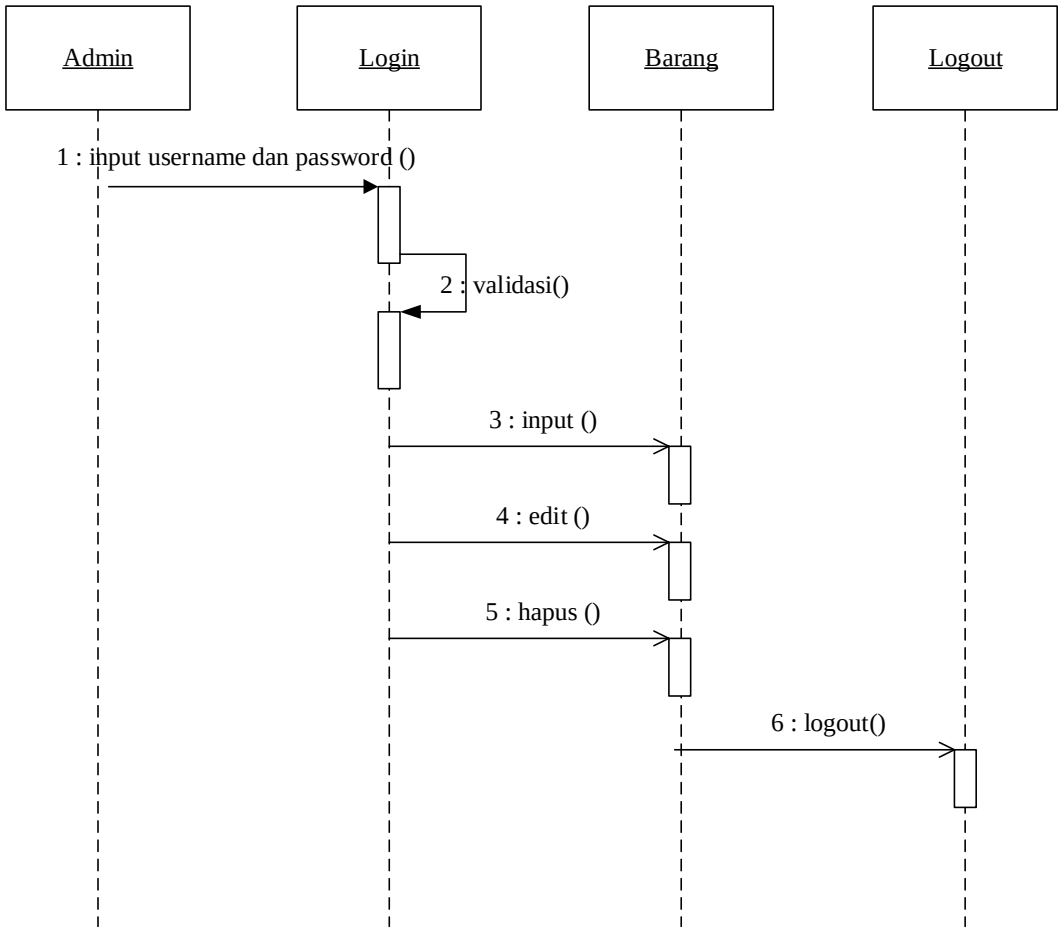


Gambar II.3. *Class* Diagram
(Sumber :Abulwafa Muhammad, et al., 2013 : 57)

3. *Sequence* Diagram

Sequence diagram digunakan untuk menggambarkan perilaku actor pada sebuah system secara detail menurut waktu. Diagram ini menunjukkan sejumlah contoh objek dan *message* (pesan) yang diletakkan diantara objek-objek di dalam *use case* (Abulwafa Muhammad, et al., 2013 : 58).

Adapun contoh Sequence Diagram dapat dilihat pada gambar II.4.



.Gambar II.4. *Sequence* Diagram
(Sumber :Abulwafa Muhammad, et al., 2013 : 58)